浮動小数 Grobner 基底の計算法 (Computer Algebra : Design of Algorithms, Implementations and Applications)

AUTHOR(S):

佐々木, 建昭; 加古, 富志雄

数理解析研究所講究録
第1568巻 2007年 142-148

浮動小数Gröbner基底の計算法

佐々木 建昭 (Tateaki Sasaki) *
筑波大学 数学系
INSTITUTE OF MATHEMATICS, UNIVERSITY OF TSUKUBA

加古 富志雄 (Fujio Kako) †
奈良女子大学 理学部
DEPARTMENT OF COMP. SCI., NARA WEMEN'S UNIVERSITY

Abstract

浮動小数係数の多項式系のGröbner基底計算は、重要であるにも拘らず、現在の計算機代数における難問の一つである。本稿では誰も考えだにしなかった方法を提案する：微小係数を記号で置き換え、有効浮動小数を利用して計算する方法である。実験の結果、期待どおりに動作することを確認した。

1 はじめに

浮動小数Gröbner基底は2種類に分類できる。第一種は、入力多項式の係数は正確あるいは任意の精度まで計算可能だが、何らかの理由で浮動小数を用いる場合。この場合、計算結果の精度が低下すれば、計算精度を上げて再計算すればよい。第二種は、入力多項式の係数が浮動小数である場合。この場合、計算中に精度が落ちると取り返しがつかないので、精度を落さないように計算を進める必要がある。本稿で扱うのは第二種のGröbner基底である。

第二種Gröbner基底に関しては二つの誤解があるようだ。第一の誤解とは、頭係数が非常に小さいときにのみ計算が不安定になるというもの。2章で例示するように、現実には頭係数が1/100 ~ 1/10であっても計算が全く不安定になることが多々ある。第二の誤解とは、浮動小数を有理数で近似すれば（時間はかかるが）計算できるというもの。浮動小数が誤差を持たず正確に有理数に変換できる場合にはこれは正しい。しかし、有理数計算でも桁落ちが生じ、誤差を持つ場合には誤差が計算結果に大きく効くので、その場合にはとんでもない答えが得られることになる。

浮動小数Gröbner基底の最初の研究者はShirayanagi [Shi93, Shi96] である。彼が扱ったのは第一種なので、我々の参考にはならない。第二種のGröbner基底はStetter[Ste97]、Fortuna,Gianni,Trager [FGT01]、Traverso,Zanoni [TZ02]、Traverso [Tra02]、Weispfenning [Wei03]、Kondratyev,Stetter,Winkler [KSW04]、Stetter [Ste05] により研究された。[Ste97] でStetterは、非常に小さい頭項を最低順位項として扱い、零次元イデアルの零点計算に使える"拡張Gröbner基底"を得た。拡張Gröbner基底は[KSW04]で研究された。[FGT01] の著者らは求める結果がSylvester行列の特異値分解で得られる特殊な場合を扱った。TraversoとZanoniは、摂動系がどう変化するかをみるため、条件 $\epsilon^2 = 0$ を満たす変数 ϵ を導入した。ϵ により系の不安定性は見えるが、Gröbner基底を安定に計算することはできない。[Tra02] でTraversoは浮動小数Gröbner基底計算に対する種々のアイデアを検討し、悲観的な見方を示している。Weispfenningは微小項を

*sasaki@math.tsukuba.ac.jp
†kako@ics.nara-wu.ac.jp

Comprehensive Gröbner 基底の枠組で扱うことを提案しているが、筆者らはこれは筋違いだと思う。[Ste05] で Stetter は幾つかの可能性を論じたが、彼の講演タイトル が現状を明白に示している。

浮動小数 Gröbner 基底の計算が困難なのは計算中に巨大な誤差が生じるからである。誤差は計算が進むとともに、$O(10^{10}) \Rightarrow O(10^{20}) \Rightarrow$ と増大し、計算結果が全く意味のないものとなるのである。このように巨大な誤差は主要項の打ち消しから生じる（桁落ち誤差）。桁落ちは数値計算では頻繁に生じる。有名なのが行列の Gauss 消去で発生する桁落ちであるが、これはピボッティングと呼ばれる簡単な操作で劇的に改善される。数式の代数的計算でも桁落ちはよく発生する。筆者らは、浮動小数による記号ニュートン法やヘンゼル構成、終結式計算などでも巨大な桁落ちが発生しうることを発見したが、桁落ちを回避する方法も見出した。ところが、Gröbner 基底計算では、桁落ちを回避する方法は未だ見出されていない。

本稿は、まず巨大な桁落ちのメカニズムを明らかにする。次に、系がその構造上示すであろう "本質的な桁落ち" を除き、無用な桁落ちを大幅に抑える方法を提案する。提案する方法は幾つかの大掛かりな工夫に基づく：各係数を "有効浮動小数" と命名した数 [KS97] で表し、微小係数を記号で置き換える。こうすることで、主要項の打ち消しによる桁落ちを回避できるのである。我々はこの方法を国産数式処理システム GAL にインプリメントし、有効に動作することを確認した。

2 桁落ちのメカニズム：クローンと自己簡約

本稿では、多項式 P の無限大ノルムを $\|P\|$ と表す。P の頭項、頭係数、残余をそれぞれ $\mathrm{ht}(P)$, $\mathrm{hc}(P)$, $\mathrm{rt}(P)$ と表す。また、$\mathrm{Spol}(P,Q)$, $\mathrm{Mred}(P,Q)$, $P \xrightarrow{Q} P'$, $P \xrightarrow{\Gamma} P'$ の意味は通常どおりである。

さて、浮動小数 Gröbner 基底計算における問題は桁落ちであることを述べたが、そのメカニズムを明らかにする。大きな桁落ちは微小主係数が現れる多項式剰余列計算でも発生する。P_1, P_2 は $\deg(P_1) \geq \deg(P_2)$ なる 1 変数多項式とし、$|\mathrm{hc}(P_1)|/\|P_1\| \gg |\mathrm{hc}(P_2)|/\|P_2\|$ とする。$P_3 = \mathrm{rem}(P_1, P_2)$ では桁落ちは起きないが、$P_4 = \mathrm{rem}(P_2, P_3)$ では巨大な桁落ちが発生する。P_2 の主項が微小なので $P_3 \approx \mathrm{const} \times \mathrm{rt}(P_2)$ となり、P_4 の計算は P_2 を（主項を除き）自分とほとんど同じ P_3 で簡約することであり、主要項をキャンセルさせる演算となっている。本稿では、$P' \approx \mathrm{const} \times \mathrm{rt}(\cdots \mathrm{rt}(P) \cdots)$ であるとき、P' は P のクローンであるといい、$P' = \mathrm{clone}(P)$ と表す。そして、多項式をそのクローンで簡約することを自己簡約と呼ぶ。

多項式剰余列は主項消去の繰り返しであり、Gröbner 基底計算は頭項消去の繰り返しであるので、後者においても自己簡約が巨大桁落ちの原因ではないかと推測できる。実例でこの推測を確認しよう。

例 1 下記 $\{P_1, P_2, P_3\}$ の Gröbner 基底を全次数順序で計算する（頭項のみを消去する）。

$$\begin{cases} P_1 &= x^3/10.0 + 3.0x^2y + 1.0y^2, \\ P_2 &= 1.0x^2y^2 - 3.0xy^2 - 1.0xy, \\ P_3 &= y^3/10.0 + 2.0x^2. \end{cases} \tag{2.1}$$

$\mathrm{Spol}(P_2,P_3) \xrightarrow{P_1} \xrightarrow{P_2} \xrightarrow{P_3} \xrightarrow{P_1} P_4$ （桁落ちなし）、$P_2 \xrightarrow{P_4} \xrightarrow{P_3} \xrightarrow{P_1} \xrightarrow{P_4} P_2'$ （桁落ちなし）、$\mathrm{Spol}(P_3,P_2') \xrightarrow{P_3} \xrightarrow{P_1} \xrightarrow{P_4} \xrightarrow{P_2'} \xrightarrow{P_3} P_5$ （$O(10^9)$ の桁落ち）、$P_4 \xrightarrow{P_5} \xrightarrow{P_2'} \xrightarrow{P_3} \xrightarrow{P_5} P_4'$ （$O(10^2)$ の桁落ち）、$P_2' \xrightarrow{P_4} \xrightarrow{P_3} \xrightarrow{P_5} \xrightarrow{P_4} P_2''$ （桁落ちなし）。Gröbner 基底（未簡約）として $\{P_2'', P_4', P_5\}$ が得られるが、係数の微小さからは想像もできない巨大な桁落ちが起きたことが分る。計算過程を詳しく見ると、$P_2' = \mathrm{clone}(P_4)$ であり、$\mathrm{Spol}(P_3,P_2')$ の M 簡約において P_4 のクローンが現れ、そのクローンが P_2' によって簡約された。すなわち、巨大な桁落ちは自己簡約の際に起きたのである。 ◇

筆者らは幾つかの例を調べたが、偶然に起きた一つの項における桁落ちを除き、他の全ては自己簡約によるものであり、また自己簡約では必ず対応する項がキャンセルした。論文 [SK07] では、自己簡約で桁落ちが生じることを証明し、桁落ち量を解明した。

3 浮動小数 Gröbner 基底計算に対する三つの工夫

前章より、自己簡約を避けることができれば、浮動小数 Gröbner 基底を安定に計算できると予想される。実際、多項式剰余列に基づく近似 GCD 計算では、自己簡約を避けることができる（論文 [SS07] を参照）。したがって、最初の工夫は下記である。

第一の工夫 S 多項式生成と M 簡約でクローンをチェックし、自己簡約を可能な限り回避する。

筆者らはこのアイデアを少しテストしたが、第一の工夫だけでは桁落ちを十分には避けられそうにない。そこで、さらに二つの工夫を行うが、それらは大掛かりな仕掛けを必要とする。

3.1 第二の工夫：微小係数の記号化

以下、マシンイプシロンを ε_{M} と表す。T を係数 1 の単項式とし、c_1, c_2, c_3 を浮動小数として、単項式 c_1T, c_2T, c_3T を考える。各係数の誤差を ε_i $(i = 1, 2, 3)$ とし、$|\varepsilon_i|/|c_i| = O(\varepsilon_{\mathrm{M}})$ かつ $0 < c_2 \ll c_1 = -c_3$ であるとする。c_1T, c_2T, c_3T をこの順に加えると、現在の数式処理システムでは次のように計算が行われる（次式で、c は係数を格納するメモリである）。

$$c := c_1 \implies c := c + c_2 \implies c := c + c_3. \tag{3.2}$$

最後のステップで c_1 と c_3 がキャンセルする。このキャンセルは、正確な演算では何の問題も起こさないが、浮動小数計算では第 2 ステップで c_1 の誤差が c_2 を "汚し"、そのため第 3 ステップで c_1 と c_3 がキャンセルしたあと、誤差が相対的に $O([c_1/c_2]\varepsilon_{\mathrm{M}})$ に拡大された c_2 が答えとなるのである。

この桁落ち誤差は従来の方法では不可避である。しかし、もしも項 c_2T を別扱いできるならば、下記のように桁落ち誤差を避けることができる（次式で、d は c_2 を別に格納するメモリである）。

$$c := c_1 \implies d := c_2 \implies c := 0 = c + c_3 \implies c := d = c + d. \tag{3.3}$$

ここで、期待通りの結果を得るには、第 3 ステップで $c_1 + c_3$ を正しく 0（浮動小数の 0 ではなく数学の 0）に計算する必要がある。そのための計算法は次節に述べるが、本節では項 c_2T を別扱いする工夫を述べる。ただし、全ての項を別扱いする必要はなく、クローン中の主要項以外の項を別扱いしさえすればよい。

アイデア自体は簡単である…微小係数を記号で置き換えて計算を進め、最後に記号を元の数値に戻すのである。ただし、記号係数が常に多項式になるように計算を進める。この方法によりクローン中の微小項が別扱いされ、主要項が見事にキャンセルすることを例で見よう。

例 2 微小頭係数の記号的扱いと主要項のキャンセル（全次数順序）。

$$\begin{aligned}
P_1 &= x^4y + x^2y - 2xy^2 \\
P_2 &= x^2y^3 + 2x^2y - 3xy^2 \\
P_3 &= \boxed{s}x^3y^2 + x^2y^2 - x^2y + 2xy^2 && /*\ \boxed{s}\ \text{is a small \#} \\
&\quad - \\
P_4 &= \mathrm{Spol}(P_1, P_3) = \boxed{s}yP_1 - xP_3 && /*\ P_4 = \mathrm{clone}(P_3) \\
&= -x^3y^2 + x^3y + (\boxed{s} - 2)\,x^2y^2 - 2\boxed{s}xy^3 \\
P_5 &= \mathrm{Spol}(P_2, P_3) = \boxed{s}xP_2 - yP_3 && /*\ P_5 = \mathrm{clone}(P_3) \\
&= -x^2y^3 + 2\boxed{s}x^3y - (3\boxed{s} - 1)\,x^2y^2 - 2xy^3 \\
P_6 &= \mathrm{Spol}(P_4, P_5) = yP_4 - xP_5 && /*\ \text{self-reduction} \\
&\quad (= ey^2P_1 - xyP_3 - ex^2P_2 + xyP_3) \\
&= \boxed{s} \times (-2x^4y + 3x^3y^2 + x^2y^3 - 2xy^4)
\end{aligned}$$

P_3 の頭項が微小なので、$P_4 = \mathrm{Spol}(P_1, P_3)$ と $P_5 = \mathrm{Spol}(P_2, P_3)$ は共に P_3 のクローンであり、$\mathrm{Spol}(P_4, P_5)$ は自己簡約である。P_4 と P_5 の微小項は $\boxed{s}$ に比例する項として別扱いされている。 ◇

3.2 第三の工夫：有効浮動小数の利用

前節で、誤差がなければ正確にキャンセルする項を浮動小数で0と判定する必要性を述べた。これは一見単純そうだが、実際に浮動小数で計算すると誤差のみからなる微小項が多く残り、どれを0と判定するかは単純ではない。何らかの工夫が必要であるが、我々は以前に考案した“有効浮動小数”を使う。次章で見るように、有効浮動小数は多項式の規格化にも非常に有用である。

有効浮動小数 (effective floating-point number) は、桁落ち誤差を自動的に検出するため、1997年に筆者の二人が考案した。GALでは $\#E[f,e]$ と表現され、“値部”f は通常の浮動小数演算での計算結果であり、“誤差部”e はその誤差を近似的に表す。有効浮動小数は次の演算規則で計算される。

$$\begin{aligned}
\#E[f_a,e_a]+\#E[f_b,e_b] &\Longrightarrow \#E[f_a+f_b,\ \max\{e_a,e_b\}],\\
\#E[f_a,e_a]-\#E[f_b,e_b] &\Longrightarrow \#E[f_a-f_b,\ \max\{e_a,e_b\}],\\
\#E[f_a,e_a]\times\#E[f_b,e_b] &\Longrightarrow \#E[f_a\times f_b,\ \max\{|f_be_a|,|f_ae_b|\}],\\
\#E[f_a,e_a]\div\#E[f_b,e_b] &\Longrightarrow \#E[f_a\div f_b,\ \max\{|e_a/f_b|,|f_ae_b/f_b^2|\}].
\end{aligned} \tag{3.4}$$

この演算規則が示すように、誤差部の計算では桁落ち誤差のみが考慮され、丸め誤差は無視される。

数式処理システムGALでは、$|f|<e$ のとき有効浮動小数は0と判定される。また、誤差部 e は実際の誤差より5倍程度大きく初期設定される。現実のコンピュータでは2倍長浮動小数のマシンイプシロンは $\varepsilon_{\mathrm{M}}\simeq 0.2\times10^{-15}$ なので、GALでは e は相対的に 1.0×10^{-15} に初期設定される。これらのため、二つの項がキャンセルして結果の相対誤差が 0.5×10^{-15} になったとしても、計算結果は0と判定される。実際のプログラムでは念をいれて、$|f|/e<3$ の場合には0にするチェックを時々行っている。

4 インプリメンテーションの詳細

上述の計算法をインプリメントするには種々の細かい工夫が必要である。簡単な工夫の一つは、M簡約を頭項にのみ適用することである。本章では、紙面の制約上、三つの工夫のみを記述する。

4.1 オーダ記号の導入と高次項のカットオフ

通常のGröbner基底計算では係数の除算を行うから、微小係数を記号で置き換えると有理式を扱うことになるが、我々は多項式係数として計算を行うことを述べた。具体的にはS多項式とM簡約を次式で計算する（次式では $\mathrm{ht}(P_i)=c_iT_i$ $(i=1,2)$ とし、$\mathrm{Mred}(P_1,P_2)$ では $T_2\,|\,T_1$ であるとする）。

$$\mathrm{Spol}(P_1,P_2) = c_2\frac{\mathrm{LCM}}{T_1}P_1-c_1\frac{\mathrm{LCM}}{T_2}P_2,\qquad \mathrm{LCM}=\mathrm{lcm}(T_1,T_2), \tag{4.5}$$

$$\mathrm{Mred}(P_1,P_2) = c_2P_1-c_1[T_1/T_2]P_2. \tag{4.6}$$

この計算をそのまま実行すると、計算の進行とともに係数多項式が指数関数的に膨張し、計算不能に陥る。幸いなことに、記号の高次項は数値的には微小なので、ある程度以上の項を切捨てて計算してもよい。

実際のプログラムでは、i 番目の微小係数は記号 $\#s[i]$ で置き換えるが、その際、その係数の値に応じてオーダ記号 $\#o$ の巾乗を付与する。$\#o$ はその値がおおよそ $1/10$ となるように巾を決める。具体的には、微小係数 c_i が $0.5\times10^{-k}\le|c_i|<5\times10^{-k}$ であるとき、$|c_i|$ を $\#o^k\#s[i]$ で置き換える。そして、オーダ記号 $\#o$ をべき級数変数とみなし、K 次より高い次数の項をカットする。

4.2 多項式の規格化

多項式の規格化は、浮動小数 Gröbner 基底の計算自体ではそれほど重要ではないが、近似 Gröbner 基底の計算では決定的に重要である（因みに [SK07] では近似 Gröbner 基底を論じている）。実際に浮動小数で計算を行うと、個々の式がどの程度の精度で計算されたか（どれ程の桁落ちが生じたか）を一目で知りたくなるが、以下に述べる規格化はそれに合致するものである。

有効浮動小数 $\#E[f,e]$ が計算の結果 $\#E[f',e']$ になったなら、その過程で起きた桁落ちは $|f/e| \div |f'/e'|$ で与えられる。したがって、$e = e'$ となるように規格化すれば、桁落ち量は $|f/f'|$ で与えられる。入力式の係数は相対誤差 $\varepsilon_{\text{Init}}$ を含むとし、多項式 F の係数を $\#E[f_1,e_1]$, $\#E[f_2,e_2]$, $\cdots$, $\#E[f_m,e_m]$ とする。このとき、次式を満たすように F を規格化する。

$$\max\{e_1, e_2, \cdots, e_m\} = \varepsilon_{\text{Init}}. \tag{4.7}$$

この規格化では各入力多項式 F_i は $\|F_i\| = 1$ となる。また、F 全体に生じた桁落ち量（各係数の桁落ちの最小値）は $1/\max\{|f_1|, \cdots, |f_m|\}$ で与えられる。記号係数の場合には、各係数の中で $\#o$ の位数が最小の項に着目し、それらの項の係数の誤差部に対して (4.7) を適用する。

4.3 計算制御パラメータの適応的決定

プログラムには計算を制御する重要なパラメータが二つ含まれる。一つは上述の高次項カット用のパラメータ K であり、他の一つはどの程度の微小係数を記号で置き換えるかを制御するパラメータ η である：$|c_i| < \eta$ ならば係数 c_i を記号 $\#s[i]$ で置き換える。K の値は桁落ちが大きいほど大きく選ぶ必要がある。一方、η の値を小さく選ぶと桁落ちを十分には除けないだろうし、大きく選ぶと多数の係数が記号化されて計算に時間がかかる。K と η の最適値は、特に K の最適値は計算を実行してみないと分らない。

我々の戦略は、最初にデフォルト値で試験的に計算し、それを見て最適値を決めるというものである。

5 実験

上述の計算法を GAL の Gröbner 基底パッケージにインプリメントした。因みに、GAL は有効浮動小数を装備している。上述の計算法と従来の計算法を比較するため、実験例は有理数体上でも正確に計算できるものを選び、S 多項式生成での対選択と M 簡約の順番は有理数体上の計算法と同じにして実験を行った（したがって、自己簡約の回避策は全く入っていない）。計算結果の精度は有理数体上での計算結果と比較して決めた。計算制御パラメータのデフォルト値は $K = 30, \eta = 0.1$ とした。

実験結果の数値の記述について：有効浮動小数の値部が 3 桁以下の正確な数であればそのまま記述するが、4 桁以上の場合は上位 4 桁のみを記述する。誤差部は上位 3 桁のみを記述する。GAL は有効浮動小数 $\#E[1.0,\ 1.0\text{e}{-15}]$ を 1 と出力するので、それをそのまま記述する。

例 3 例 1 の系を規格化し、記号係数法で計算する（$K = 30$）。

$$\begin{cases} P_1 &= \#E[0.03333, 3.33\text{e}{-17}]\, x^3 + x^2y + \#E[0.3333, 3.33\text{e}{-16}]\, y^2, \\ P_2 &= \#E[0.3333, 3.33\text{e}{-16}]\, x^2y^2 - xy^2 - \#E[0.3333, 3.33\text{e}{-16}]\, xy, \\ P_3 &= \#E[0.05, 5.0\text{e}{-17}]\, y^3 + x^2. \end{cases}$$

まず、$\text{hc}(P_3)$ と $\text{hc}(P_1)$ をそれぞれ記号 $\#s[1]$, $\#s[2]$ で置き換える（プログラムが勝手に行う）。

$$\begin{aligned} P_3 &= y^3(\#o^2\#s[1]) + x^2, \\ P_1 &= x^3(\#o^2\#s[2]) + x^2y + \#E[0.3333, 3.33\text{e}{-16}]\, y^2. \end{aligned}$$

途中結果を省略し、最後に記号を数値に戻す直前の計算結果を部分的に示す。

$$\begin{aligned}
P_5 &= x^2(-\#E[0.003921, 3.13\text{e}{-17}]\#o^{10}\#s[2]^5 + \cdots \text{ (up to } \#o^{24})) \\
&+ xy(-\#E[0.1058, 5.29\text{e}{-16}]\#o^{10}\#s[2]^4\#s[1] + \cdots \text{ (up to } \#o^{24})) \\
&+ y^2(-\#E[0.01176, 1.0\text{e}{-15}]\#o^{10}\#s[2]^4\#s[1] + \cdots \text{ (up to } \#o^{24})), \\
P_4' &= xy(+\#E[0.1072, 5.29\text{e}{-16}]\#o^{12}\#s[2]^5\#s[1] + \cdots \text{ (up to } \#o^{30})) \\
&+ y^2(+\#E[0.01206, 1.0\text{e}{-15}]\#o^{12}\#s[2]^5\#s[1] + \cdots \text{ (up to } \#o^{30})), \\
P_2'' &= y^2(-\#E[0.000001838, 1.0\text{e}{-15}]\#o^{14}\#s[2]^5\#s[1]^2 + \cdots \text{ (up to } \#o^{30})).
\end{aligned}$$

P_5, P_4', P_2'' はコンテントとしてそれぞれ $\#o^8\#s[2]^4$, $\#o^{10}\#s[2]^5$, $\#o^{10}\#s[2]^5$ を持つ。このことは、これらの多項式の計算において、それぞれ 10^8, 10^{10}, 10^{10} 程度の桁落ちが起きたことを示している。最後に、記号 $\#s[1]$ と $\#s[2]$ に対応する数値を代入し、頭係数を 1 に規格化すると次式が得られた。

$$\begin{aligned}
P_2'' &= y^2, \\
P_4' &= xy + \#E[\underline{0.08440225504518}, 9.84\text{e}{-15}]\,y^2, \\
P_5 &= x^2 + \#E[\underline{7.1484968974627}, 3.69\text{e}{-14}]\,xy \\
&\quad + \#E[\underline{0.57371613952465}, 6.97\text{e}{-14}]\,y^2.
\end{aligned}$$

下線部は正しい数値を示す（P_4' の第二係数の下 3 桁 518 は有理数体上の計算では 522 となる）。

我々は、$K = 25$ として計算してみたが、この場合は下 4 ～ 5 桁は誤差となった。また、微小係数 0.0333$\cdots$ と 0.05 を同一記号で置き換えて計算してみた（現在のプログラムは大きさの違いが 2 以下の数値は同一の記号で置き換えるオプションを持つ）が、結果にはほとんど差がなかった。 ◇

例 4 下記の系 $\Gamma_0 = \{P_1, P_2, P_3\}$ の Gröbner 基底を記号係数法で計算する。

$$\begin{cases}
P_1 &= \#E[0.01, 1.0\text{e}{-17}]\,x^3 + x^2y + y^2, \\
P_2 &= x^2y^2 - xy^2 + y, \\
P_3 &= xy^2 + \#E[0.01, 1.0\text{e}{-17}]\,y^3 + x^2.
\end{cases}$$

本例では、係数の記号化をいつ行うか、二つの選択肢がある：第一の選択肢は、微小係数が頭項に現れた時点で記号化するものであり、第二の選択肢は、頭項以外の微小係数も記号化するものである。前者では、まず $\mathrm{hc}(P_1)$ が記号化され、次に $P_2 \xrightarrow{P_3} \xrightarrow{P_3} y^4/10000 - x^3 + x^2y/100 - xy^2 + y$ と M 簡約され、そのあと簡約結果式の頭係数が記号化される。この簡約結果式をみれば、P_3 の微小係数が二つの項に伝搬していることが分り、したがってより多くの微小係数を記号化しなければならないことになる。記号化される係数が多いほど計算量は飛躍的に増大するので、我々は第一の選択肢を採用する。

P_1 と P_2 の微小係数を同一記号 $\#s[1]$ で置き換えると、次式が得られる。

$$\begin{aligned}
P_1 &= x^3(\#o^2\#s[1]) + x^2y + y^2, \\
P_3 &= xy^2 + y^3(\#o^2\#s[1]) + x^2.
\end{aligned}$$

この記号化では、たとえば $\mathrm{Spol}(P_3, P_1)$ は次の式となる。

$$\begin{aligned}
P_4 &= y^5(\#o^4\#s[1]^2 - \#o^8\#s[1]^4) + x^4(-\#o^2\#s[1]) \\
&+ x^3y(-1 + \#o^4\#s[1]^2) + x^2y^2(\#o^2\#s[1] - \#o^6\#s[1]^3) + y^4.
\end{aligned}$$

途中の計算式は省略し、最後の結果だけを記述する（$K = 25$ で計算した）。

$$\begin{aligned}
P_6 &= y, \\
P_5' &= x^2 - \#E[\underline{0.00019606833239274}, 1.18\text{e}{-18}]\,y^2 \\
&\quad + \#E[\underline{0.029510774413745}, 1.78\text{e}{-16}]\,y.
\end{aligned}$$

下線は正しい数値を示す。望み通りに Gröbner 基底が精度良く計算できた。 ◇

6 本質的な桁落ちについて

例3で、最終結果の P_5 には約120の桁落ちが発生しているが、この桁落ちは自己簡約によるものではない(計算過程で1/3の大きさの頭項が現れるが、それは自己簡約を引き起こさない)。$P_5 = a_1P_1 + a_2P_2 + a_3P_3$ なる a_1, a_2, a_3 を計算すると、$\|a_1P_1\| = \|a_3P_3\| = 65.44$, $\|a_2P_2\| = 21.81$ となる。すなわち、P_5 における桁落ちの大部分は本質的なもので、どのように計算しようとも生じるものである。この本質的な桁落ちは非常に大きくなる場合がある。実際、筆者等は2000にもなる例を見つけている。このように大きな本質的桁落ちの存在は、浮動小数 Gröbner 基底計算にも "悪条件" な系があることを示している。

別の型の本質的桁落ちは近似 Gröbner 基底の計算で現れる。近似 Gröbner 基底の計算では M 簡約の結果が微小になる多項式を0と近似して計算を進めることになろう(具体的には論文 [SK07] を参照)。$P \xrightarrow{\Gamma}\!\!\!\!\rightarrow \tilde{P}$, $\|\tilde{P}\|/\|P\| = \varepsilon \ll 1$ となる場合、この計算過程では全体として $O(1/\varepsilon)$ の桁落ちが生じるが、それは本質的桁落ちである。上述のように悪条件な系もあるので、桁落ちが系の悪条件性によるのか、あるいは近似性にあるのか、見極める必要がある。

参 考 文 献

[FGT01] E. Fortuna, P. Gianni and B. Trager: Degree reduction under specialization. J. Pure Appl. Algebra, 164(1-2):153-164, 2001.

[KS97] F. Kako and T. Sasaki: Proposal of "effective" floating-point number. May 1997 (unpublished).

[KSW04] A. Kondratyev, H.J. Stetter and S. Winkler: Numerical computation of Gröbner bases. Proc. CASC 2004, 295-306, St. Petersburg, Russia, 2004.

[Shi93] K. Shirayanagi: An algorithm to compute floating-point Gröbner bases. Mathematical Computation with Maple V: Ideas and Applications, T. Lee (Ed.), Birkhäuser, 95-1-6, 1993.

[Shi96] K. Shirayanagi: Floating point Gröbner bases. Mathematics and Computers in Simulation **42** (1996), 509-528.

[SK07] T. Sasaki F. Kako: Computing floating-point Gröbner base stably. preprint of Univ. Tsukuba, Jan. 2007.

[SS07] M. Sanuki and T. Sasaki: Computing approximate GCD's in ill-conditioned cases. Preprint of Univ. Tsukuba, Jan. 2007.

[Ste97] H.J. Stetter: Stabilization of polynomial systems solving with Gröbner bases. Proc. ISSAC'97, 117-124, 1997, ACM Press.

[Ste05] H.J. Stetter: Approximate Gröbner bases – an impossible concept? Proc. SNC 2005,235-236, 2006; full paper will appear in *Symbolic-Numeric Computations (Trends in Mathematics)*, D. Wand and L. Zhi (Eds.), Birkhäuser Verlag, 2007.

[Tra02] C. Traverso: Syzygies, and the stabilization of numerical Buchberger algorithm. Proc. LMCS 2002, 244-255, RISC-Linz.

[TZ02] C. Traverso and A. Zanoni: Numerical stability and stabilization of Gröbner basis computation. Proc. ISSAC 2002, 262-269, 2002, ACM Press.

[Wei03] V. Weispfenning: Gröbner bases for inexact input data. Proc. CASC 2003, 403-411, Passau, Germany, 2003.